USING SIMPLE NUMERICAL SCHEMES TO COMPUTE VISUAL FEATURES WHENEVER UNAVAILABLE

Application to a Vision-Based Task in a Cluttered Environment

FOLIO David and CADENAT Viviane
LAAS/CNRS, 7 avenue du Colonel ROCHE,Toulouse, France
Paul SABATIER University, 118 route de Narbonne, Toulouse, France

Keywords: Visual features estimation, visual servoing, collision avoidance.

Abstract: In this paper, we address the problem of estimating image features whenever they become unavailable during a vision-based task. The method consists in using numerical algorithm to compute the lacking data and allows to treat both partial and total visual features loss. Simulation and experimental results validate our work for two different visual-servoing navigation tasks. A comparative analysis allows to select the most efficient algorithm.

1 INTRODUCTION

Visual servoing techniques aim at controlling the robot motion using visual features provided by a camera and require that image features remain always visible (Corke, 1996; Hutchinson et al., 1996). However, different problems may occur during the execution of a given task: visual features loss or occlusions, camera failure, and so on. In such cases, the above mentioned techniques cannot be used anymore and the corresponding task will not be achieved. Thus the visual features visibility during a vision-based task appears as an interesting and challenging problem which must be addressed. Classically, the proposed solutions aim at *avoiding* occlusions and loss. Most of these solutions are dedicated to manipulator arms because such robotic systems offer a sufficient number of degrees of freedom (DOF) to benefit from redundancy to treat this kind of problem (Marchand and Hager, 1998; Mansard and Chaumette, 2005). Other techniques preserve visibility by path-planning in the image (Mezouar and Chaumette, 2002), by acting on specific DOFs (Corke and Hutchinson, 2001; Malis et al., 1999; Kyrki et al., 2004), by controlling the zoom (Benhimane and Malis, 2003) or by making a tradeoff with the nominal vision-based task (Remazeilles et al., 2006). In a mobile robotic context, when executing a vision-based navigation task in a cluttered environment, it is necessary to preserve not only the visual features visibility but also the robot safety. A first answer to this double problem has been proposed in (Folio and Cadenat, 2005a; Folio and Cadenat, 2005b). The developed methods allow to avoid collisions, occlusions and target loss when executing a vision-based task amidst obstacles. However they are restricted to missions where it is possible to avoid both occlusions and collisions without leading to local minima. Therefore, a true extension of these works would be to provide methods which accept that occlusions may effectively occur. A first solution is to allow *some* of the features to appear and disappear temporarily from the image as done in (Garcia-Aracil et al., 2005). However, this approach is limited to partial occlusions. Another solution which is considered in this paper is to compute the visual features as soon as some or all of them become unavailable. Total visual features loss can then be specifically treated.

The paper is organized as follows. In section 2, we propose a method allowing to compute the visual features when they become unavailable. Section 3 describes the application context, and shows some simulation and experimental results validating our work.

2 VISUAL DATA ESTIMATION

In this section, we address the visual data estimation problem whenever they become unavailable. We first introduce some preliminaries and state the problem

FOLIO D. and CADENAT V. (2007).
USING SIMPLE NUMERICAL SCHEMES TO COMPUTE VISUAL FEATURES WHENEVER UNAVAILABLE - Application to a Vision-Based Task in a Cluttered Environment.
In *Proceedings of the Fourth International Conference on Informatics in Control, Automation and Robotics*, pages 326-332
DOI: 10.5220/0001628503260332

before presenting our estimation method.

2.1 Preliminaries

We consider a mobile camera and a vision-based task with respect to a static landmark. We assume that the camera motion is holonomic, characterized by its kinematic screw: $\mathcal{T}_c = (V_{C/\mathcal{F}_0}^{\mathcal{F}_c T}, \Omega_{\mathcal{F}_c/\mathcal{F}_0}^{\mathcal{F}_c T})^T$ where $V_{C/\mathcal{F}_0}^{\mathcal{F}_c} = (V_{X_c}, V_{Y_c}, V_{Z_c})^T$ and $\Omega_{\mathcal{F}_c/\mathcal{F}_0}^{\mathcal{F}_c} = (\Omega_{X_c}, \Omega_{Y_c}, \Omega_{Z_c})^T$ represent the translational and rotational velocity of the camera frame $\mathcal{F}_c$ with respect to the world frame $\mathcal{F}_0$ expressed in $\mathcal{F}_c$ (see figure 1).

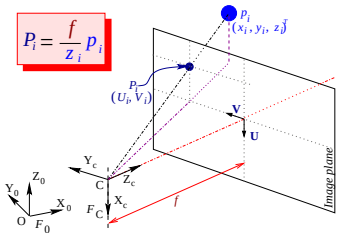


Figure 1: The pinhole camera model.

Remark 1 *We do not make any hypothesis about the robot on which is embedded the camera. Two cases may occur: either the robot is holonomic and so is the camera motion; or the robot is not, and we suppose that the camera is able to move independantly from it.*

Now, let s be a set of visual data provided by the camera, and $\mathbf{z}$ a vector describing its depth. For a fixed landmark, the variation of the visual signals $\dot{s}$ is related to $\mathcal{T}_c$ by means of the interaction matrix $\mathcal{L}$ as shown below (Espiau et al., 1992):

$$\dot{s} = \mathcal{L}(s, \mathbf{z})\, \mathcal{T}_c \qquad (1)$$

This matrix allows to link the visual features motion in the image to the 3D camera motion. It depends mainly on the depth $\mathbf{z}$ (which is not always available on line) and on the considered visual data. We suppose in the sequel that we will only use image features for which (1) can be computed analytically. Such expressions are available for different kinds of features such as points, straight lines, circles (Espiau et al., 1992), or image moments (Chaumette, 2004)...

2.2 Problem Statement

Now, we focus on the problem of estimating (*all* or some) visual data s whenever they become unavailable. Different approaches, such as tracking methods and signal processing techniques, may be used to deal with this kind of problem. Here, we have chosen to use a simpler approach for several reasons. First of all, most tracking algorithms relies on measures from the image which is unavailable in our case. Second, as it is intended to be used to perform complex navigation tasks, the estimated visual signals must be provided sufficiently rapidly with respect to the control law sampling period. Finally, in our application, the initial value of the visual features to be estimated is always known, until the image become unavailable. Thus, designing an observer or a filter is not necessary, as this kind of tools is mainly interesting when estimating the state of a dynamic system whose initial value is unknown. Another idea is to use a 3D model of the object together with projective geometry in order to deduce the lacking data. However, this choice would lead to depend on the considered landmark type and would require to localize the robot. This was unacceptable for us, as we do not want to make any assumption on the visual feature model. Therefore, we have chosen to solve numerically equation (1) on the base of the visual signals previous measurements and of the camera kinematic screw $\mathcal{T}_c$.

As a consequence, our method will lead to closed loop control schemes for any task where the camera motion is defined independently from the image features. This will be the case for example when executing a vision-based task in a cluttered environment if the occlusion occurs in the obstacle neighborhood, as shown in section 3. However, for "true" vision-based tasks where $\mathcal{T}_c$ is directly deduced from the estimated visual signals, the obtained control law remains an open-loop scheme. Therefore, it will be restricted to occlusions of short duration and when there is small perturbation on the camera motion. Nonetheless, in the context of a sole visual servoing navigation task, this approach remains interesting as it appears as an *emergency* solution when there is no other mean to recover from a complete loss of the visual data. This method can also be used to predict the image features between two image acquisitions. It is then possible to compute the control law with a higher sampling period, and it is also quite interesting in our context.

Now, let us state our problem. As equation (1) depends on depth $\mathbf{z}$, it is necessary to evaluate this parameter together with the visual data s. Therefore, our method requires to be able to express analytically the variation $\mathbf{z}$ with respect to the camera motion. Let us suppose that $\dot{\mathbf{z}} = \mathcal{L}_{\mathbf{z}} \mathcal{T}_c$. Using relation (1) and denoting by $\mathbf{X} = (s^T, \mathbf{z}^T)^T$, the differential equations to be solved can be expressed as:

$$\begin{cases} \dot{\mathbf{X}} & = & \begin{pmatrix} \mathcal{L}^T & \mathcal{L}_{\mathbf{z}}{}^T \end{pmatrix}^T \mathcal{T}_c = \mathbf{F}(\mathbf{X}, t) \\ \mathbf{X}(t_0) & = & \mathbf{X}_0 = (s_0^T, \mathbf{z}_0^T)^T \end{cases} \qquad (2)$$

where $\mathbf{X}_0$ is the initial value of $\mathbf{X}$, which can be considered known as s_0 is directly given by the feature extraction processing and $\mathbf{z}_0$ can be usually charac-

terized off-line. Finally, for the problem to be well stated, we assume that $\mathcal{T}_c$ has a very small variation during each integration step $h = t_{k+1} - t_k$ of (2).

We propose to use numerical schemes to solve the Ordinary Differential Equations (ODE) (2). A large overview of such methods is proposed for example in (Shampine and Gordon, 1975). In this work, our objective is to compare several numerical schemes (Euler, Runge-Kutta, Adams-Bashforth-Moulton (ABM) and Backward Differentiation Formulas (BDF)) to select the most efficient technique.

3 APPLICATION

We have chosen to apply the considered numerical algorithms in a visual servoing context to compute the visual features when they are lost or unavailable during a navigation task. We have considered two kinds of missions: the first one is a positioning vision-based task during which a camera failure occurs; the second one is a more complex mission consisting in realizing a visually guided navigation task amidst obstacles despite possible occlusions and collisions.

After describing the robotic system, we present the two missions and the obtained results.

3.1 The Robotic System

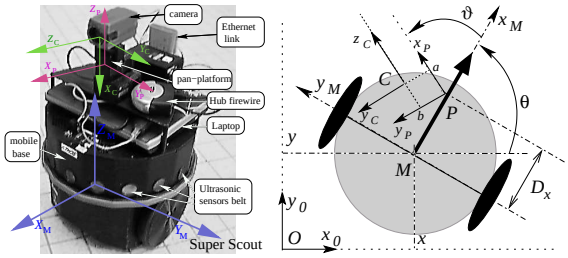
We consider the mobile robot SuperScout II[1] equipped with a camera mounted on a pan-platform (see figure 2(a)). It is a small cylindric cart-like vehicle, dedicated to indoor navigation. A DFW-VL500 Sony color digital IEEE1394 camera captures pictures in YUV 4:2:2 format with 640480 resolution. An image processing module allows to extract points from the image. The robot is controlled by an on-board laptop computer running under Linux on which is installed a specific control architecture called G^{en}oM (Generator of Module) (Fleury and Herrb, 2001). When working on the robot, three different sampling periods are involved:

1. T_E : the integration step defined by $h = t_{k+1} - t_k$,
2. $T_{Ctrl} \simeq 44$ms : the control law sampling period,
3. $T_{Sens} \simeq 100$ms : the camera sampling period.

As Linux is not a real-time OS, these values are only approximatively known.

First, let us model our system to express the camera kinematic screw. To this aim, consider figure 2(b). (x,y) are the coordinates of the robot reference point M with respect to the world frame $\mathcal{F}_O$.

[1]The mobile robot SuperScout II is provided by the AIP-PRIMECA.



(a) Nomadic SuperScout II. (b) Modelisation.

Figure 2: The robotic system.

θ and ϑ are respectively the direction of the vehicle and the pan-platform with respect to the x_M-axis. P is the pan-platform center of rotation, D_x the distance between M and P. We consider the successive frames: $\mathcal{F}_M\ (M, x_M, y_M, z_M)$ linked to the robot, $\mathcal{F}_P\ (P, x_P, y_P, z_P)$ attached to the pan-platform, and $\mathcal{F}_c\ (C, x_c\ , y_c\ , z_c)$ linked to the camera. The control input is defined by the vector $\dot{q} = (v, \omega, \varpi)^T$, where v and ω are the cart linear and angular velocities, and ϖ is the pan-platform angular velocity with respect to $\mathcal{F}_M$. For this specific mechanical system, the kinematic screw is related to the joint velocity vector by the robot jacobian $\mathbf{J}$: $\mathcal{T}_c = \mathbf{J}\dot{q}$. As the camera is constrained to move horizontally, it is sufficient to consider a reduced kinematic screw $\mathcal{T}_r = (V_{y_c}, V_{z_c}, \Omega_{x_c})^T$, and a reduced jacobian matrix $\mathbf{J}_r$ as follows:

$$\mathcal{T}_r = \begin{pmatrix} -\sin(\vartheta) & D_x\cos(\vartheta)+a & a \\ \cos(\vartheta) & D_x\sin(\vartheta)-b & -b \\ 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} v \\ \omega \\ \varpi \end{pmatrix} = \mathbf{J}_r\,\dot{q} \quad (3)$$

As $\det(\mathbf{J}_r) = D_x \neq 0$, so the jacobian $\mathbf{J}_r$ is always invertible. Moreover, as the camera is mounted on a pan-platform, its motion is holonomic (see remark 1).

3.2 Execution of a Vision-Based Task Despite Camera Failure

Our objective is to perform a vision-based task despite a camera failure. We first describe the considered mission and state the estimation problem for this particular task before presenting the obtained results.

The Considered Vision-based Task. Our goal is here to position the embedded camera with respect to a landmark made of n points. To this aim, we have applied the visual servoing technique given in (Espiau et al., 1992) to mobile robots as in (Pissard-Gibollet and Rives, 1995). In this approach which relies on the task function formalism (Samson et al., 1991), the visual servoing task is defined as the regulation to zero of the following error function:

$$e_{VS}(q,t) = \mathcal{C}\,(s(q,t) - s^*) \quad (4)$$

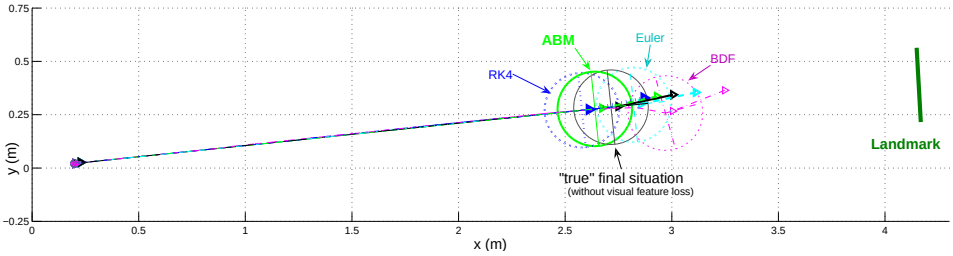


Figure 3: Robot trajectories obtained for the different schemes.

where s is a $2n$-dimensional vector made of the coordinates (U_i, V_i) of each 3D projected point P_i in the image plane (see figure 1). s^* is the desired value of the visual signal, while $\mathcal{C}$ is a full-rank combination matrix which allows to take into account more visual features than available DOF (Espiau et al., 1992). Classically, a kinematic controller, $\dot{q}_{\text{VS}}$ can be determined by imposing an exponential convergence of e_{VS} to zero: $\dot{e}_{\text{VS}} = C\mathcal{L}\mathbf{J}_{\text{r}}\dot{q}_{\text{VS}} = -\lambda_{\text{VS}} e_{\text{VS}}$, where λ_{VS} is a positive scalar or a positive definite matrix. Fixing $\mathcal{C} = \mathcal{L}^+$ as in (Espiau et al., 1992), we get:

$$\dot{q}_{\text{VS}} = \mathbf{J}_{\text{r}}^{-1}(-\lambda_{\text{VS}})\mathcal{L}^+(s(q,t) - s^*) \qquad (5)$$

where $\mathcal{L}$ is a $2n3$ matrix deduced from the classical optic flow equations (Espiau et al., 1992) as follows:

$$\mathcal{L}_i(P_i, z_i) = \begin{bmatrix} 0 & \frac{U}{z_i} & \frac{U_i V_i}{f} \\ -\frac{f}{z_i} & \frac{V_i}{z_i} & f + \frac{V_i^2}{f} \end{bmatrix} \text{ with } i = 1..n \qquad (6)$$

where f is the camera focal.

Estimation Problem Statement. Let us state our estimation problem by defining the expression of the ODE (2) in the case of the considered task. As we consider a target made of n points, we need first to determine the depth variation of each of these points. It can be easily shown that, for one 3D point p_i of coordinates $(x_i, y_i, z_i)^T$ in $\mathcal{F}_c$ projected into a point $P_i(U_i, V_i)$ in the image plane as shown in figure 1, the depth variation $\dot{z}_i$ is related to the camera motion according to: $\dot{z}_i = \mathcal{L}_{z_i}\mathcal{T}_{\text{r}}$, with $\mathcal{L}_{z_i} = [0 \quad -1 \quad \frac{z_i}{f} V_i]$. Thus, for the considered task, the ODE to be solved for one point P_i are given by:

$$\begin{cases} \dot{U}_i = \frac{U_i}{z_i} V_{Z_c} + \frac{U_i V_i}{f} \Omega_{X_c} \\ \dot{V}_i = -\frac{f}{z_i} V_{Y_c} + \frac{V_i}{z_i} V_{Z_c} + \left(f + \frac{V_i^2}{f}\right) \Omega_{X_c} \\ \dot{z}_i = -V_{Z_c} - \frac{z_i}{f} V_i \Omega_{X_c} \end{cases} \qquad (7)$$

Finally, for the considered landmark made of n points, the ODE (2) are deduced from the above relation (7) by defining: $\mathbf{X} = [U_1 \; V_1 \; \ldots \; U_n \; V_n, \; z_1 \; \ldots \; z_n]^T$.

Experimental Results. We have experimented the considered vision-based task on our mobile robot SuperScout II. For each numerical scheme, we have performed the same navigation task: start from the same configuration using the same $s^\star$. At the beginning of the task, $\dot{q}_{\text{VS}}$ uses the visual features available from the camera and the robot starts converging towards the target. At the same time, the numerical algorithms are initialized and launched. After 10 steps, the landmark is artificially occluded to simulate a camera failure and, if nothing is done, it is impossible to perform the task. Controller (5) is then evaluated using the computed values provided by each of our numerical algorithms and the robot is controlled by an open-loop scheme. For each considered numerical scheme figure 3 shows the robot trajectories and table 1 summarizes the whole results. These errors remain small, which means that there are few perturbations on the system and, in this case, our "emergency" open-loop control scheme allows to reach a neighborhood of the desired goal despite the camera failure. Moreover, *for the proposed task*, the ABM scheme is the most efficient method, as it leads to the least standard deviation error (std) and to the smallest maximal error. The RK4 algorithm gives also correct performances, while Euler method remains the less accurate scheme as expected. As T_{E} is rather small, the BDF technique provides correct results but has been proven to be much more efficient when there are sudden variations in the kinematic screw as it will be shown in the next part.

Table 1: Results synthesis.

Schemes	s / z	std error	max error
Euler	s (pix)	1.0021	9.6842
	z (m)	0.088256	0.72326
RK4	s (pix)	0.90919	7.0202
	z (m)	0.07207	0.63849
ABM	s (pix)	**0.90034**	**5.9256**
	z (m)	**0.05721**	**0.50644**
BDF	s (pix)	1.1172	7.6969
	z (m)	0.10157	0.5989

3.3 Realizing a Navigation Task Amidst Possibly Occluding Obstacles

Our goal is to realize a positioning vision-based task amidst possibly occluding obstacles. Thus, two prob-

lems must be addressed: the visual data loss and the risk of collision. The first one will be treated using the above estimation technique and the second one thanks to a rotative potential field method. We describe the control strategy before presenting the results.

Collision and Occlusion Detection. Our control strategy relies on the detection of the risks of collision and occlusion. The danger of collision is evaluated from the distance d_{coll} and the relative orientation α between the robot and the obstacle deduced from the US sensors mounted on the robot. We define three envelopes around each obstacle ξ_+, ξ_0, ξ_-, located at $d_+ > d_0 > d_-$ (see figure 4). We propose to model the *risk of collision* by parameter μ_{coll} which smoothly increases from 0 when the robot is far from the obstacle ($d_{\text{coll}} > d_0$) to 1 when it is close to it ($d_{\text{coll}} < d_-$).

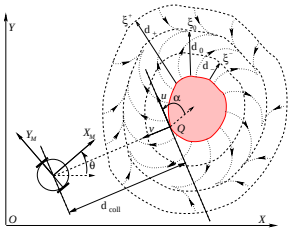
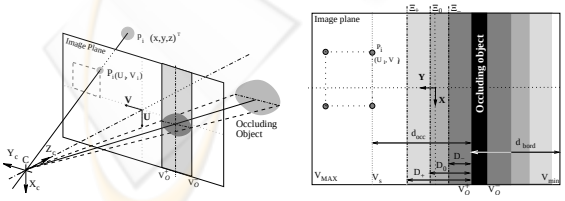


Figure 4: Obstacle avoidance.

The occlusion risk is evaluated from the detection of the occluding object left and right borders extracted by our image processing algorithm (see figure 5(a)). From them, we can deduce the shortest distance d_{occ} between the image features and the occluding object O, and the distance d_{bord} between O and the opposite image side to the visual features. Defining three envelopes Ξ_+, Ξ_0, Ξ_- around the occluding object located at $D_+ > D_0 > D_-$ from it, we propose to model the risk of occlusion by parameter μ_{occ} which smoothly increases from 0 when O is far from the visual features ($d_{\text{occ}} > D_0$) to 1 when it is close to them ($d_{\text{occ}} < D_-$). A possible choice for μ_{coll} and μ_{occ} can be found in (Folio and Cadenat, 2005a).



(a) Occluding object projection in the image plane.

(b) Definition of the relevant distances in the image.

Figure 5: Occlusion detection.

Global Control Law Design. Our global control strategy relies on μ_{coll} and μ_{occ}. It consists in two steps. First we define two controllers allowing respectively to realize the sole vision-based task and to guarantee non collision while dealing with occlusions in the obstacle vicinity. Second, we switch between these two controllers depending on the risk of occlusion and collision. We propose the following global controller:

$$\dot{q} = (1-\mu_{\text{coll}})\dot{q}_{\text{VS}} + \mu_{\text{coll}}\dot{q}_{\text{coll}} \tag{8}$$

where $\dot{q}_{\text{VS}}$ is the visual servoing controller previously defined (5), while $\dot{q}_{\text{coll}} = (v_{\text{coll}}\ \omega_{\text{coll}}\ \varpi_{\text{coll}})^T$ handles obstacle avoidance and visual signal estimation if necessary. Thus, when there is no risk of collision, the robot is driven using only $\dot{q}_{\text{VS}}$ and executes the vision-based task. When the vehicle enters the obstacle neighborhood, μ_{coll} increases to reach 1 and the robot moves using only $\dot{q}_{\text{coll}}$. This controller is designed so that the vehicle avoids the obstacle while tracking the target, treating the occlusions if any. It is then possible to switch back to the vision-based task once the obstacle is overcome. The avoidance phase ends when both visual servoing and collision avoidance controllers point out the same direction: $sign(\dot{q}_{\text{VS}}) = sign(\dot{q}_{\text{coll}})$, and if the target is not occluded ($\mu_{\text{occ}} = 0$). In this way, we benefit from the avoidance motion to make the occluding object leave the image.

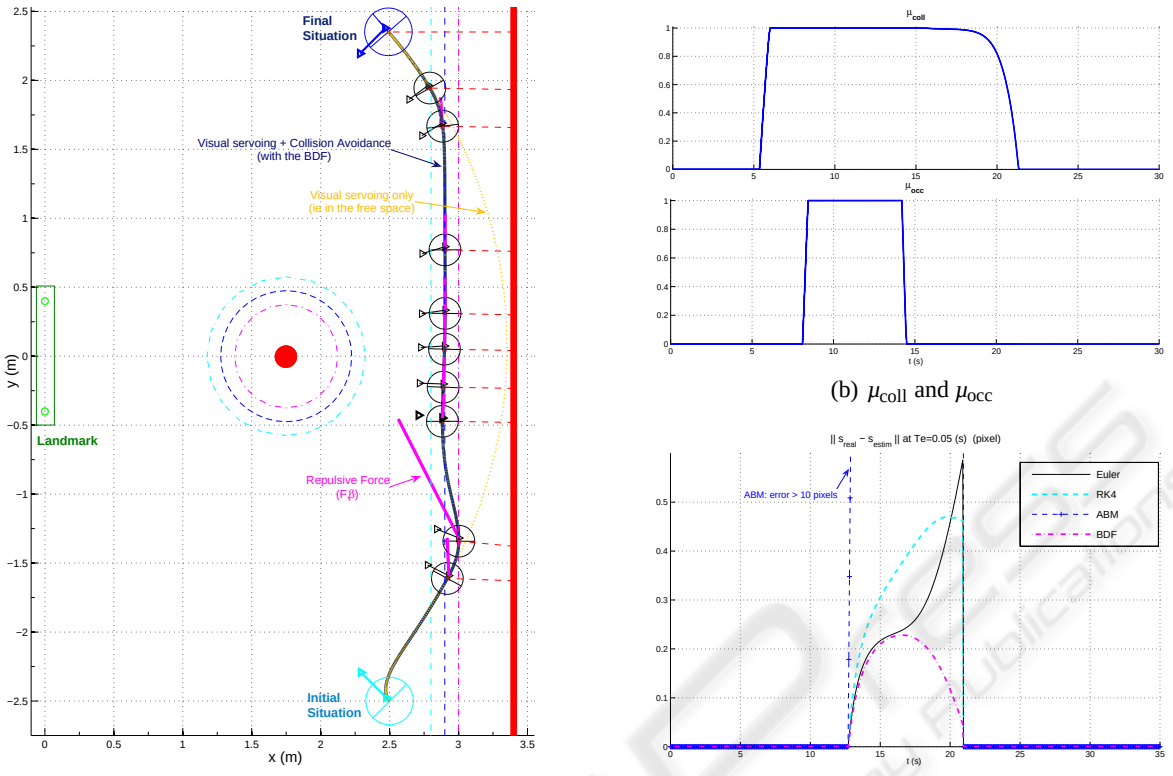
Remark 2 *Controller* (8) *allows to treat occlusions which occur during the avoidance phase. However, obstacles may also occlude the camera field of view without inducing a collision risk. In such cases, we may apply to the robot either another controller allowing to avoid occlusions as done in (Folio and Cadenat, 2005a; Folio and Cadenat, 2005b) for instance, or the open-loop scheme based on the computed visual features given in section 3.2.*

Now, it remains to design $\dot{q}_{\text{coll}}$. We propose to use a similar approach to the one used in (Cadenat et al., 1999). The idea is to define around each obstacle a rotative potential field so that the repulsive force is orthogonal to the obstacle when the robot is close to it ($d_{\text{coll}} < d_+$), parallel to the obstacle when the vehicle is at a distance d_0 from it, and progressively directed towards the obstacle between d_0 and d^+ (as shown on figure 4). The interest of such a potential is that it can make the robot move around the obstacle without requiring any attractive force, reducing local minima problems. We use the same potential function as in (Cadenat et al., 1999):

$$\begin{cases} U(d_{\text{coll}}) = \frac{1}{2}k_1\left(\frac{1}{d_{\text{coll}}} - \frac{1}{d^+}\right)^2 + \frac{1}{2}k_2(d_{\text{coll}} - d^+)^2 & \text{if } d_{\text{coll}} \leq d^+ \\ U(d_{\text{coll}}) = 0 & \text{otherwise} \end{cases} \tag{9}$$

where k_1 and k_2 are positive gains to be chosen. v_{coll} and ω_{coll} are then given by (Cadenat et al., 1999):

$$\dot{q}_{\text{b}} = \begin{pmatrix} v_{\text{coll}} & \omega_{\text{coll}} \end{pmatrix}^T = \begin{pmatrix} k_v F\cos\beta & \frac{k\omega}{D_x} F\sin\beta \end{pmatrix}^T \tag{10}$$



(a) Robot trajectory (using BDF)

(b) μ_{coll} and μ_{occ}

(c) Error $\|s - \widetilde{s}\|$ (pixel) for the different schemes

Figure 6: Simulation results.

where $F = -\frac{\partial U}{\partial d_{\text{coll}}}$ is the modulus of the virtual repulsive force and $\beta = \alpha - \frac{\pi}{2d_0} d_{\text{coll}} + \frac{\pi}{2}$ its direction with respect to $\mathcal{F}_M$. k_v and k_ω are positive gains to be chosen. Equation (10) drives only the mobile base in the obstacle neighborhood. However, if the pan-platform remains uncontrolled, it will be impossible to switch back to the execution of the vision-based task at the end of the avoidance phase. Therefore, we have to address the ϖ_{coll} design problem. Two cases may occur in the obstacle vicinity: either the visual data are available or not. In the first case, the proposed approach is similar to (Cadenat et al., 1999) and the pan-platform is controlled to compensate the avoidance motion while centering the target in the image. As the camera is constrained to move within an horizontal plane, it is sufficient to regulate to zero the error $e_{\text{gc}} = V_{\text{gc}} - V_{\text{gc}}^*$ where V_{gc} and $V_{\text{gc}}^\star$ are the current and desired ordinates of the target gravity center. Rewriting equation (3) as $\mathcal{T}_{\text{r}} = J_{\text{b}}\dot{q}_{\text{b}} + J_\varpi \varpi_{\text{coll}}$ and imposing an exponential decrease to regulate e_{gc} to zero ($\dot{e}_{\text{gc}} = \mathcal{L}_{V_{\text{gc}}} \mathcal{T}_{\text{r}} = -\lambda_{\text{gc}} e_{\text{gc}}$, $\lambda_{\text{gc}} > 0$), we finally obtain (see (Cadenat et al., 1999) for more details):

$$\varpi_{\text{coll}} = \frac{-1}{\mathcal{L}_{V_{\text{gc}}} J_\varpi} (\lambda_{\text{gc}} e_{\text{gc}} + \mathcal{L}_{V_{\text{gc}}} J_{\text{b}} \dot{q}_{\text{b}}) \qquad (11)$$

where $\mathcal{L}_{V_{\text{gc}}}$ is the 2$^{\text{nd}}$ row of $\mathcal{L}_i$ evaluated for V_{gc} (see equation (6)). However, if the obstacle occludes the camera field of view, s is no more available and the pan-platform cannot be controlled anymore using (11). At this time, we compute the visual features by integrating the ODE (2) using one of proposed numerical schemes. It is then possible to keep on executing the previous task e_{gc}, even if the visual features are temporary occluded by the encountered obstacle. The pan-platform controller during an occlusion phase will then be deduced by replacing the real target gravity center ordinate V_{gc} by the computed one $\widetilde{V}_{\text{gc}}$ in (11). We get:

$$\widetilde{\varpi}_{\text{coll}} = \frac{-1}{\widetilde{\mathcal{L}}_{V_{\text{gc}}} J_\varpi} (\lambda_{\text{gc}} \widetilde{e}_{\text{gc}} + \widetilde{\mathcal{L}}_{V_{\text{gc}}} J_{\text{b}} \dot{q}_{\text{b}}), \qquad (12)$$

where $\widetilde{e}_{\text{gc}} = \widetilde{V}_{\text{gc}} - V_{\text{gc}}^*$ and $\widetilde{\mathcal{L}}_{V_{\text{gc}}}$ is deduced from (6). Now, it remains to apply the suitable controller to the pan-platform depending on the context. Recalling that the parameter $\mu_{\text{occ}} \in [0;1]$ allows to detect occlusions, we propose the following avoidance controller:

$$\dot{q}_{\text{coll}} = \begin{pmatrix} v_{\text{coll}}, & \omega_{\text{coll}}, & (1-\mu_{\text{occ}})\varpi_{\text{coll}} + \mu_{\text{occ}} \widetilde{\varpi}_{\text{coll}} \end{pmatrix}^T \qquad (13)$$

Simulation Results. The proposed method has been simulated using Matlab software. We aim at positioning the camera with respect to a given landmark despite two obstacles. D_-, D_0 and D_+ have been fixed to

40, 60 and 115 pixels, and d_-, d_0, d_+ to 0.3m, 0.4m, and 0.5m. For each numerical scheme (Euler, RK4, ABM and BDF), we have performed the same navigation task, starting from the same situation and using the same $s^\star$. Figure 6(c) shows that the BDF are the most efficient scheme, while ABM is the worst, RK4 and Euler giving correct results for this task. Indeed, as the obstacle avoidance induces important variations in the camera motion, ODE (2) becomes stiff, and the BDF have been proven to be more suitable in such cases. Figures 6(a) and 6(b) show the simulation results obtained using this last scheme. The task is perfectly performed despite the wall and the circular obstacle. The different phases of the motion can be seen on the evolution of μ_{coll} and μ_{occ}. At the beginning of the task, there is no risk of collision, nor occlusion, and the robot is driven by $\dot{q}_{\text{VS}}$. When it enters the wall neighborhood, μ_{coll} increases and $\dot{q}_{\text{coll}}$ is applied to the robot which follows the security envelope ξ_0 while centering the landmark. When the circular obstacle enters the camera field of view, μ_{occ} increases and the pan-platform control smoothly switches from ϖ_{coll} to $\widetilde{\varpi}_{\text{coll}}$. It is then possible to move along the security envelope ξ_0 while tracking a "virtual" target until the end of the occlusion. When there is no more danger, the control switches back to $\dot{q}_{\text{VS}}$ and the robot perfectly realizes the desired task.

4 CONCLUSIONS

In this paper, we have proposed to apply classical numerical integration algorithms to determine visual features whenever unavailable during a vision-based task. The obtained algorithms have been validated both in simulation and experimentation with interesting results. A comparative analysis has been performed and has shown that the BDF is particularly efficient when ODE (2) becomes stiff while giving correct results in more common use. Therefore, it appears to be the most interesting scheme.

REFERENCES

Benhimane, S. and Malis, E. (2003). Vision-based control with respect to planar and non-planar objects using a zooming camera. In *Int. Conf. on Advanced Robotics*, Coimbra, Portugal.

Cadenat, V., Souères, P., Swain, R., and Devy, M. (1999). A controller to perform a visually guided tracking task in a cluttered environment. In *Int. Conf. on Intelligent Robots and Systems*, Korea.

Chaumette, F. (2004). Image moments: a general and useful set of features for visual servoing. *Trans. on Robotics and Automation*, 20(4):713–723.

Corke, P. (1996). *Visual control of robots : High performance visual servoing*. Research Studies Press LTD.

Corke, P. and Hutchinson, S. (2001). A new partitioned approach to image-based visual servo control. *Trans. on Robotics and Automation*, 17:507–515.

Espiau, B., Chaumette, F., and Rives, P. (1992). A new approach to visual servoing in robotics. *Trans. on Robotics and Automation*.

Fleury, S. and Herrb, M. (2001). *G^{en}oM : User Manual*. LAAS-CNRS.

Folio, D. and Cadenat, V. (2005a). A controller to avoid both occlusions and obstacles during a vision-based navigation task in a cluttered environment. In *European Control Conference(ECC-CDC'05)*.

Folio, D. and Cadenat, V. (2005b). Using redundancy to avoid simultaneously occlusions and collisions while performing a vision-based task amidst obstacles. In *European Conference on Mobile Robots*, Ancona, Italy.

Garcia-Aracil, N., Malis, E., Aracil-Santonja, R., and Perez-Vidal, C. (2005). Continuous visual servoing despite the changes of visibility in image features. *Trans. on Robotics and Automation*, 21.

Hutchinson, S., Hager, G., and Corke, P. (1996). A tutorial on visual servo control. *Trans. on Robotics and Automation*.

Kyrki, V., Kragic, D., and Christensen, H. (2004). New shortest-path approaches to visual servoing. In *Int. Conf. on Intelligent Robots and Systems*.

Malis, E., Chaumette, F., and Boudet, S. (1999). 2 1/2d visual servoing. *Trans. on Robotics and Automation*, 15:238–250.

Mansard, N. and Chaumette, F. (2005). A new redundancy formalism for avoidance in visual servoing. In *Int. Conf. on Intelligent Robots and Systems*, volume 2, pages 1694–1700, Edmonton, Canada.

Marchand, E. and Hager, G. (1998). Dynamic sensor planning in visual servoing. In *Int. Conf. on Robotics and Automation*, Leuven, Belgium.

Mezouar, Y. and Chaumette, F. (2002). Avoiding self-occlusions and preserving visibility by path planning in the image. *Robotics and Autonomous Systems*.

Pissard-Gibollet, R. and Rives, P. (1995). Applying visual servoing techniques to control a mobile hand-eye system. In *Int. Conf. on Robotics and Automation*, Nagoya, Japan.

Remazeilles, A., Mansard, N., and Chaumette, F. (2006). Qualitative visual servoing: application to the visibility constraint. In *Int. Conf. on Intelligent Robots and Systems*, Beijing, China.

Samson, C., Leborgne, M., and Espiau, B. (1991). *Robot Control. The Task Function Approach*, volume 22 of *Oxford Engineering Series*. Oxford University Press.

Shampine, L. F. and Gordon, M. K. (1975). *Computer Solution of Ordinary Differential Equations*. W. H. Freeman, San Francisco.